(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

A4

(19) World Intellectual Property Organization
International Bureau

PCT

(43) International Publication Date
11 October 2001 (11.10.2001)

(10) International Publication Number
WO 01/76133 A1

(51) International Patent Classification[7]: H04L 9/08

(21) International Application Number: PCT/GB01/00765

(22) International Filing Date: 22 February 2001 (22.02.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
00302741.4 31 March 2000 (31.03.2000) EP

(71) Applicant *(for all designated States except US)*: BRITISH TELECOMMUNICATIONS PUBLIC LIMITED COMPANY [GB/GB]; 81 Newgate Street, London EC1A 7AJ (GB).

(72) Inventor; and
(75) Inventor/Applicant *(for US only)*: BRISCOE, Robert, John [GB/GB]; Home Farm, Parham, Woodbridge, Suffolk IP13 9NW (GB).

(74) Agent: WILSON, Peter, David: BT Group Legal Services, Intellectual Property Department, Holborn Centre, 8th floor, 120 Holborn, London EC1N 2TE (GB).

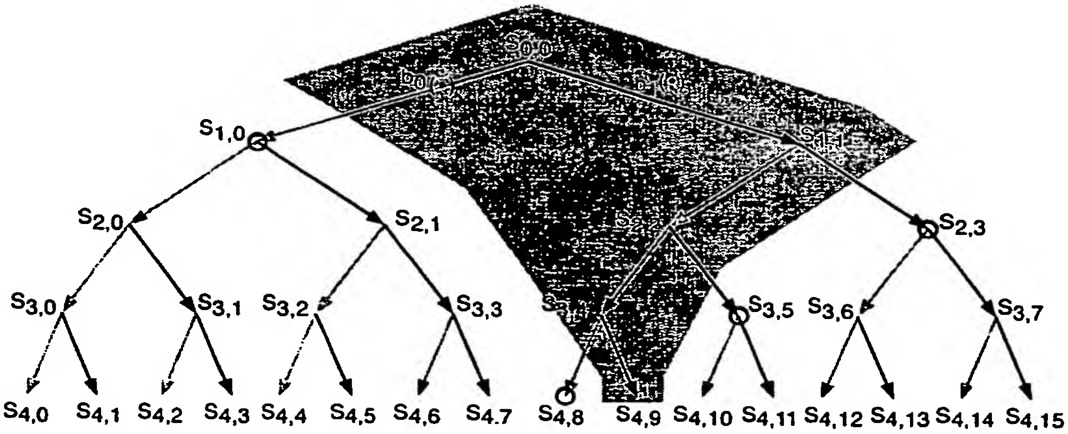
(81) Designated States *(national)*: AU, CA, JP, US.

(84) Designated States *(regional)*: European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR).

Published:
— *with international search report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

(54) Title: DATA DISTRIBUTION



(57) Abstract: We present a highly efficient method for re-keying a large group with a highly volatile membership. Each new key is calculated using a root-seeded one-way function tree with each new member being given the seeds to produce their own individually masked copy of the tree. However, the approach is wide open to attack by collusion between an attacker an any legitimate member of the group. Nevertheless, we show that all comparable schemes in the literature, despite their claims, are equally prone to such collusion whilst also being less efficient. This leads us to suggest that the focus of the research agenda should move to the far more difficult problem of copy detection or prevention, whether for data or key material.

WO 01/76133 A1

DATA DISTRIBUTION

This invention relates to a method of distributing data packets to as plurality of users over a communications network, and in particular to a method of multicasting streams of data packets over the internet

When multicasting data to a number of users over a communications network it is often important that the multicast data packets are only delivered to authorised users. For example, a subscription may be required to receive the data packets or commercially sensitive information may be being transmitted across a VPN intranet.

When using Internet multicast, senders send to a multicast group address while receivers 'join' the multicast group through a message to their local router. For scalability, the designers of IP multicast deliberately ensured that any one router in a multicast tree would hide all downstream join and leave activity from all upstream routers and senders [S. Deering, "Multicast Routing in a Datagram Network," PhD thesis, Dept. of Computer Science, Stanford University, (1991).]. Thus a multicast sender is oblivious to the identities of its receivers. Clearly any security relationship with individual receivers is impossible if they can't be uniquely distinguished. Conversely, if receivers have to be distinguished from each other, the scalability benefits start to be eroded. The same dilemma arises with other bulk data distribution methods, such as satellite broadcast or distribution of mass-produced identical physical media such as digital versatile disks (DVDs). The only known solution to this dilemma is to encrypt the data delivered in bulk to anyone interested, then restrict the circulation of a key to decrypt it. Thus the cost of bulk data distribution can be kept low, by keeping it openly available and uncustomised, while a relatively small amount of data can be targeted more carefully at some identifying feature of each legitimate customer in order to unlock the larger bulk.

The present invention provides a technique to maintain the secrecy of data sent to a group of receivers with a highly volatile membership which is more efficient than any in the known literature. The approach is theoretically vulnerable to attack by collusion between an attacker and any legitimate member of the group. Nonetheless, it can be shown that all comparable schemes in the literature are equally prone to collusion as they rely on the colluders being 'well-behaved' - only leaking the mostly strongly fortified aspect of each sch me, rather than the weakest point - usually the shared group key itself. The only exceptions are approaches where the data is transcoded by intermediaries on every data path, which tend to negate the efficiency gains of data distribution technologies, such as satellite broadcast or Internet multicast. Thus, we show it is misleading to quote the collusion resistance of a key management scheme in isolation. As key management is only one contributor to the collusion resistance of closed group data distribution, it need be no stronger against collusion than the weakest link in the chain.

If a sender to a group wishes to restrict its data to a set of receivers, it will typically encrypt the data at the application level. End-to-end access is then controlled by limiting the circulation of the key. A new receiver could have been storing away the encrypted stream before it joined the secure session. Therefore, every time a receiver is allowed in, the key needs to be changed (termed backward security [McGrew, David A., & Alan T. Sherman, "Key establishment in large dynamic groups using one-way function trees," TIS Report No. 0755, TIS Labs at Network Associates, Inc., Glenwood, MD (May 1998)]). Similarly, after a receiver is thrown out or requests to leave, it will still be able to decrypt the stream unless the key is changed again (forward security).

A 'secure multicast session' is defined as the set of data that a receiver *could* understand, having passed one access control test. If one key is used for many related multicast groups, they all form one secure session. If a particular receiver leaves a multicast group then re-joins but she could have decrypted the information she missed, the whole transmission is still a single secure session. We envisage very large receiver communities, e.g. ten million viewers for a popular Internet pay-TV channel. Even if just 10% of the audience tuned in or out within a fifteen minute period, this would potentially cause thousands of secure joins or leaves per second.

We use the term 'application data unit' (ADU) as a more general term for the minimum useful atom of data from a security or commercial point of view (one second in the above example). The ADU equates to the aggregation interval used in Chang *et al* "Key Management for Secure Internet Multicast using Boolean Function Minimization Techniques", Proceedings IEEE Infocomm'99, Vol2 689-698 (Mar 1999) has also been called a cryptoperiod when measured in units of time. ADU size is application and security scenario dependent. It may be an initialisation frame and its set of associated 'P-frames' in a video sequence or it may be ten minutes of access to a network game. Note that the ADU from a security point of view can be different from that used at a different layer of the application. For performance, an ADU may only be partially encrypted with the remainder in the clear [T Kunkelmann et al (Darmstadt Tech Uni), "Evaluation of Different Video Encryption Methods for a Secure Multimedia Conferencing Gateway", 4th COST 237 Workshop, Lisboa, Portugal, Springer Verlag LNCS 1356, ISBN 3-540-63935-7 (Dec 1997)]. ADU size can vary throughout the duration of a stream dependent on the content. ADU size is a primary determinant of system scalability. If a million receivers were to join within fifteen minutes, but the ADU size was also fifteen minutes, this would only require one re-key event.

However, reduction in re-keying requirements isn't the only scalability issue. In the above example, a system that can handle a million requests in fifteen minutes still has to be provided, even if its output is just one re-key request to the senders. With just such scalability problems in mind, many multicast key management architectures introduce a key manager role as a separate concern from the senders. This deals with policy concerns over membership and isolates the senders from much of the messaging traffic needed for access requests.

Re-multicast of received data requires very low resources on the part of any receiver. Even if the value of the information received is relatively low there is always a profit to be made by re-multicasting data and undercutting the original price (arbitrage), as proved in Herzog *et al*, "Sharing the cost of Multicast Trees: An Axiomatic Analysis", in Proceedings of ACM/SIGCOMM '95, Cambridge, MA, Aug. 1995. In general, prevention of information copying is considered infeasible; instead most attention focuses on the more tractable problem of copy detection. It is possible to 'watermark' different copies of a copyrighted digital work. If a watermarked copy is later discovered, it can be traced back to its source, thus deterring the holders of original copies from passing on further, illicit copies. Watermarks are typically applied to the least significant bits of a medium to avoid significantly degrading the quality. Such bits are in different locations with different regularity in different media, therefore there is never likely to be a generic approach.

Naor *et al* "Threshold Traitor Tracing", CRYPTO '98, formalises a pragmatic approach to 'traitor tracing' by proposing a parameter that represents the minimum number of group members that need to collude to eliminate a watermark. The elimination criteria are that none of the conspirators are identifiable, and it is assumed that the copyright owner will want to avoid accusing innocent members.

Beyond the requirements that have been focussed on so far, two taxonomies of multicast security requirements include many other possible combinations of security requirements for multicast, for example sender authentication (see Balenson *et al* "Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization", February 26, 1999, and Canetti *et al*, "Multicast Security: A Taxonomy and Efficient Constructions", Proceedings IEEE Infocomm'99, Vol2 708-716 (Mar 1999)) and proof of delivery.

According to a first aspect of the invention there is provided a method of distributing data to a plurality of users, each user being identified by a unique user identifier and each user being associated with a unique seed value that is generated from a hierarchical tree of seed values, the method comprising the steps of;

(a) encrypting a plurality of data units each with a group key;

(b) communicating one of said data units encrypted with the associated group key to the plurality of users;

(c) communicating subsequent encrypted data units to the plurality of users, the group key being updated in response to changes in the plurality of users; and

(d) instructing the plurality of users to update the group key accordingly,

each user storing a respective subset of the hierarchical tree, the subset comprising one or more seed values and excluding the seed value of that user and all of the direct ancestors of that seed value.

Preferably, the hierarchical tree of seed values is generated from a hierarchical tree of blinded seeds and is created from an initial seed value such that the lowest hierarchical level of the tree comprises at least as many seeds as users. Each user may store a respective subset of the hierarchical tree comprising the minimum number of seed values necessary to define the subset of the hierarchical tree.

According to a second aspect of the invention there is provided a method of adding one or more new users to a plurality of users to which data is being distributed as described above, in which the current group key, a unique terminal identifier and a unique seed value are transmitted to each new user.

According to a third aspect of the invention there is provided a method of adding one or more new users to a plurality of users to which data is being as described above, in which: the updated group key, a unique terminal identifier and a unique seed value are transmitted to each new user; and an instruction is sent to all the other users to update the group key.

According to a fourth aspect of the invention there is provided a method of removing one or more users from a plurality of users to which data is being transmitted in accordance with any of claims 1 to 3, comprising the steps of:

(a) identifying to the plurality of users the unique terminal identifier of each user to be removed;

(b) those terminals not being removed from the plurality of users generating an ejection function dependent upon the data received in step (a); and

(c) calculating the updated group key, the calculation being dependent upon the ejection function generated in step (b).

The ejection function may be generated from the unique seed value of each user to be removed and preferably the ejection function is generated from the minimum number of seed values necessary to define the one or more users to be removed. The minimum number of seed values necessary to define the one or more users to be removed may comprise one or more seed values not associated with a user. The ejection of the user(s) may substantially coincide with the end of a data unit. Preferably the ejection function is generated by XOR-ing the seed value(s).

The group key may be blinded from the previous group key, with the ejection function being used to initialise the blinding function.

According to a fifth aspect of the invention there is provided a user terminal for use with any of the methods described above, comprising; a data unit receiver module; a data storage module for storing

(i) the unique terminal identifier, and

(ii) a subset of the hierarchical tree; and

a data processing module for

(a) decrypting received data units,

(b) updating keys, and

(c) calculating an ejection function from the stored subset of the hierarchical tree.

Preferably a smart card comprises the data storage module and/or the data processing module.

Methods embodying the present invention will now be described in further detail, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a schematic depiction of the generation of a seed value;

Figure 2 is a schematic depiction of a masked root seeded one-way function tree;

Figure 3 is a schematic depiction of a masked root seeded one-way function tree showing the selection of a number of users to be ejected from the tree; and

Figure 4 is a schematic depiction of a masked root seeded one-way function tree with uneven growth patterns.

The following notation will be used in the discussion below.

- `b(v)` is the notation used for a function that blinds the value of `v`. That is, a computationally limited adversary cannot find `v` from `b(v)`. An example of a blinding or one-way function is a hash function such as the MD5 hash [RL Rivest, "The MD5 Message-Digest Algorithm", Request for Comments (RFC) 1321, Internet Engineering Task Force (1992)] or the standard Secure Hash 1 [FIPS Publication 180-1, "Secure hash standard", NIST, U.S. Department of Commerce, Washington, D.C. (April 1995).]. Good hash functions typically require only lightweight computational resources. Hash functions are designed to reduce (or increase) an input of any size to a fixed size output. In all cases, we will use an input that is already the same size as the output, merely using the blinding property of the hash, not the size reduction property.

- $c(v_1, v_2, \ldots)$ is a function that combines the values of v_1 , v_2 etc. such that given the result and all but one of the operands, the remaining operand can be trivially deduced. $c()$ should also be chosen such that, if the bits of the operands are independent and unbiased, the bits of the result will also be independent and unbiased. The XOR function is a simple example of such a combinatorial function. {$c()$ should also ideally be the function that can be used to trivially deduce the remaining operand, as is the case with XOR, that is: $v_1 = c(\ c(v_1, v_2, \ldots), v_2, \ldots)$.}

and

$o(x)$ is notation for 'of order x'.

$\lfloor j/P \rfloor$ is notation for the value of j/P rounded down to the nearest integer (the floor function).

$j \bmod P$ is notation for the remainder of j/P.

The re-keying protocol requires all members to share a copy of a one-way binary tree of deterministic pseudo-random values. That is, given the root of the tree, all nodes and leaves in the tree can be calculated. However, the tree is designed so that, given any node, only nodes and leaves further from the root can feasibly be calculated. Each member has a user identity (UID) which determines which leaf of the tree 'belongs' to that user. The whole tree is revealed to each member, except the branch down to that member's leaf, which is masked off. We now describe the construction of the tree, which enables the masked tree to be described very succinctly.

A cryptographically strong pseudo-random bit generator function $b()$ is required that deterministically stretches a w-bit-wide seed value, s, into a value, s', twice the width of s. We then define two functions $b_0(s)$ and $b_1(s)$ which, given s, output the left and right halves of s' respectively. We will term these the 'left' and the 'right' blinding functions, as illustrated in Fig 1. As a concrete example, if $s_{0,0}$ is 64b wide and $b()$ is the

MD5 hash function, $b(s_{0,0})$ will output a 128b value. The most significant 64b then form $s_{1,0}$, while the remaining 64b form $s_{1,1}$.

The sender can calculate the value of any node or leaf of the RSOFT as follows:

1. The sender randomly generates an initial seed value, $s(0,0)$.
2. The sender decides on the required maximum tree depth, D (which will lead to a maximum group size, $n_0=2^D$ unless more sophisticated schemes are used to grow the tree).
3. The sender generates two 'left' and 'right' first level intermediate seed values, applying respectively the 'left' and the 'right' blinding functions to the initial seed:

   $s(1,0)=b_0(s(0,0))$; $s(1,1)=b_1(s(0,0))$.

   The sender generates four second level intermediate seed values: $s(2,0)=b_0(s(1,0))$; $s(2,1)=b_1(s(1,0))$; $s(2,2)=b_0(s(1,1))$; $s(2,3)=b_1(s(1,1))$,

   and so on, creating a binary tree of intermediate seed values to a depth of D levels.

   Formally, if $s(d,i)$ is an intermediate seed that is d levels below the initial seed, $s(0,0)$:

   $$s(d,i) = b_p(s(d-1, \lfloor i/2 \rfloor)) \qquad (3.1.1)$$

   where $p=0$ for even i and $p=1$ for odd i

Goldreich *et al* ["How to Construct Random Functions", Journal of the ACM 33(4):792-807 (Oct 1986)] prove that, given an initial random seed and assuming one way functions exist, such a construction implements a 'polyrandom' function. A polyrandom function is one whose outputs cannot be distinguished from those of a random function by any probabilistic polynomial-time algorithm. Thus, given any member UID, n, as input, the tree construction can be used to predictably output the polyrandom number $s(D,n)$. Effectively, each bit in turn of the UID determines whether to run the left or right one way function, which can be envisaged as following the left or right branch of the tree ending up at the relevant leaf. The function clearly only requires $\log D$ operations of the one way function.

When new members join the multi-cast group a e-keying protocol is required.

1. The sender randomly generates the first group key, k_0, and is then ready to start multicasting data encrypted with this key.

2. If the sender delegates key management, it must privately communicate the group key, k_0, and the initial seed, $s(0,0)$ to the key managers. We now describe how the group key is renewed as members join and are ejected from the group.

When a key manager accepts a new member's request to join the secure group, it assigns her a unique UID and records this fact. It then calculates the complements of each of the $\log D$ seeds on the branch to her leaf. The set of complementary seeds is unicasted to her privately along with the current group key. Figure 2 shows an example group, where member UID 9 is assigned to the new member, and the ringed seeds are the complementary seeds unicast to her. Thus, the masked RSOFT (MRSOFT) shown backed in light grey is revealed to her. The nodes backed in darker grey on the branch to the new member's UID remain masked from her. Of course, the key manager may prefer to cache the higher nodes in the tree, rather than continually re-calculate them.

1. As new members join, the group key should regularly be blinded from its previous value to ensure backward security. Typically, it is announced to the group (using a reliable multicast transport) that a *backward* security re-key will occur at a certain sequence number in the data. At this point all key managers, senders and receivers calculate the new group key, $k_{i+1}=b(k_i)$.

2. As members leave (or are ejected from) the group, to ensure forward security the group key must again be changed regularly. Two types of announcements are required to the group for this: i) a declaration of the subset of member UIDs to be evicted and ii) the sequence number in the data when the *forward* security re-key will take effect. Both messages can be open, but must be authenticated and use reliable transport (which may simply be regular repetition over a period of notice, but many reliable transports may be used). The two messages may be combined, but could be separate (for instance, the re-key instruction could be implicitly announced by incrementing a key index in the data header).

   Note that any unassigned UIDs are 'don't care' values with respect to ejection. Thus if all the UIDs between two ejected members are 'don't care', all of them can be ejected at the same time to make the description of the ejection more succinct.

3. One way to succinctly describe the list of member UIDs to eject is to list the indices of the nodes that are the highest common parents of the UIDs to eject. Fig 3 shows an example where member UIDs 2, 3, 9 and 13 and 15 are to be ejected, but 14 is unassigned, so can also be ejected. The highest parents to all these members are ringed in the figure and can be calculated by all members. None of

the ejected members should know *all* of the values of the seeds at these nodes, while all of the remaining members know them all.

4. All parties remaining in the secure group then combine this set of seeds (typically XORing them all together). We shall call the resulting value s_e.

All remaining parties can then calculate the new group key, defined as a value blinded from the previous key, using a blinding function b() initialised by s_e. Typically, this function might be a keyed such as HMAC [Bellare et al, "Message Authentication using Hash Functions - The HMAC Construction", RSA Laboratories CryptoBytes, 2(1), (Spring 1996) and H. Krawczyk et al "HMAC: Keyed-Hashing for Message Authentication," Request for Comments (RFC) 2104, Internet Engineering Task Force (Feb. 1997),], specifically HMAC-MD5. Formally, the new key, $k_{i+1}=b(k_i, s_e)$.

No attempt is made to change the values in the tree that are known by any leaving member, as they are useless without also knowing the group key before any future re-key. There is no point (other than minor inconvenience) designing a scheme to protect future group keys from an attacker that is required to be any more secret from the attacker than the future group keys! This statement seems innocently obvious, but it is important to the present invention, as the known literature is considerably misguided on this point.

Note that, although same notation, b(), has been used for the three blinding functions used in the protocol, in practice each could be specified to use a different function. We have suggested using MD5 to generate the tree and to re-key for backward security, and HMAC-MD5 to re-key for forward security. The choice of blinding function depends on the relative importance of performance over security - for instance MD5 and Sha-1 are the most widely used functions, and although MD5 performs faster, it has been shown to be susceptible to collision search attacks

The more members there are to be evicted, the more likely it is that their UIDs are contiguous and can be described more succinctly, either using notation for contiguous ranges, or simply listing the highest common parents. However, this will depend upon the growth of the tree to cope with unknown future membership (see Figure 4).

Generally, the more random values that are needed to build a tree, the more it can contain sustained attacks to within the bounds of the sub-tree created from each new random seed. However, for long-running sessions, there is a trade-off between security and the convenience of a continuous key-space. The randomness of the randomly generated seeds is another potential area of weakness that must be correctly designed.

All the constructions are vulnerable to collusion between valid group members. If a sub-group of members agree amongst themselves to each buy a different range of the key space, they can all share the seeds they are sent so that they can all access the union of their otherwise separate key spaces. Arbitrage is a variant of member collusion that has already been discussed. This is where one group member buys the whole key sequence then sells portions of it more cheaply than the selling price, still making a profit if most keys are bought by more than one customer. Finally, the total system security for any particular application clearly depends on the strength of the security used when setting up the session. As always, the overall security of an application using any of the constructions is as strong as the weakest part.

This weakness against collusion means that such security schemes are best implemented within a smart card, or some other form of tamper-resistant processor.

CLAIMS

1. A method of distributing data to a plurality of users, each user being identified by a unique user identifier and each user being associated with a unique seed value that is generated from a hierarchical tree of seed values, the method comprising the steps of;

(a) encrypting a plurality of data units each with a group key;

(b) communicating one of said data units encrypted with the associated group key to the plurality of users;

(c) communicating subsequent encrypted data units to the plurality of users, the group key being updated in response to changes in the plurality of users; and

(c) instructing the plurality of users to update the group key accordingly,

each user storing a respective subset of the hierarchical tree, the subset comprising one or more seed values and excluding the seed value of that user and all of the direct ancestors of that seed value.

2. A method of distributing data to a plurality of users in which the hierarchical tree of seed values is generated from a hierarchical tree of blinded seeds and is created from an initial seed value such that the lowest hierarchical level of the tree comprises at least as many seeds as users

3. A method of distributing data to a plurality of users according to claim 1 or claim 2, wherein each user stores a respective subset of the hierarchical tree comprising the minimum number of seed values necessary to define the subset of the hierarchical tree.

4. A method of adding one or more new users to a plurality of users to which data is being distributed in accordance any of claims 1 to 3, in which the current group key, a unique terminal identifier and a unique seed value are transmitted to each new user.

5. A method of adding one or more new users to a plurality of users to which data is being distributed in accordance any of claims 1 to 3, in which:

the updated group key, a unique terminal identifier and a unique seed value are transmitted to each new user; and

an instruction is sent to all the other users to update the group key.

6. A method of removing one or more users from a plurality of users to which data is being transmitted in accordance with any of claims 1 to 3, comprising the steps of:

(a) identifying to the plurality of users the unique terminal identifier of each user to be removed;

(b) those terminals not being removed from the plurality of users generating an ejection function dependent upon the data received in step (a); and

(c) calculating the updated group key, the calculation being dependent upon the ejection function generated in step (b).

7. A method of removing one or more users from a plurality of users according to claim 6, wherein the ejection function is generated from the unique seed value of each user to be removed.

8. A method of removing one or more users from a plurality of users according to claim 6, wherein the ejection function is generated from the minimum number of seed values necessary to define the one or more users to be removed.

9. A method of removing one or more users from a plurality of users according to claim 8, wherein the minimum number of seed values necessary to define the one or more users to be removed comprises one or more seed values not associated with a user.

10. A method of removing one or more users from a plurality of users according to any of claims 7 to 9, wherein the ejection of the user(s) substantially coincides with the end of a data unit.

11. A method of removing one or more users from a plurality of users according to any of claims 7 to 10, wherein the ejection function is generated by XOR-ing the seed value(s).

12. A method of removing one or more users from a plurality of users according to any of claim 6 to claim 11, wherein the group key is blinded from the previous group key, with the ejection function being used to initialise the blinding function.

13. A data carrier comprising computer code means for carrying out the method of any of the preceding claims.

14. A user terminal for use with the method of any of the claims 1 to 12, comprising;
a data unit receiver module;
a data storage module for storing
(i) the unique terminal identifier, and
(ii) a subset of the hierarchical tree;
a data processing module for
(a) decrypting received data units,
(b) updating keys, and
(c) calculating an ejection function from the stored subset of the hierarchical tree.

15. A user terminal according to claim 14, wherein a smart card comprises the data storage module and/or the data processing module.

Fig.1.

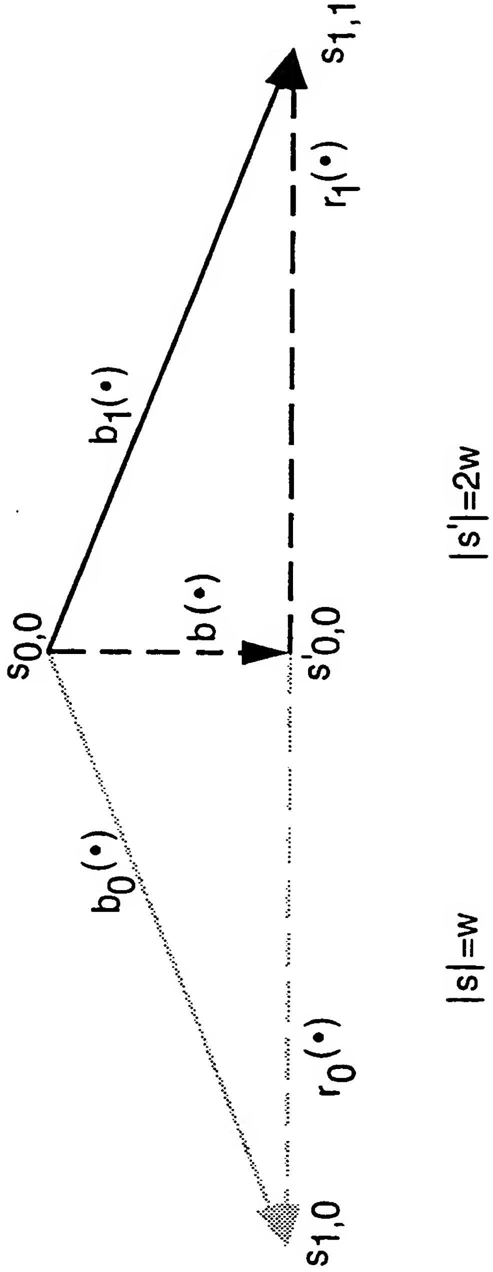

$s_{1,1}$
$r_1(\bullet)$
$b_1(\bullet)$
$|s'|=2w$
$s_{0,0}$
$b(\bullet)$
$s'_{0,0}$
$b_0(\bullet)$
$|s|=w$
$r_0(\bullet)$
$s_{1,0}$

Fig.2.

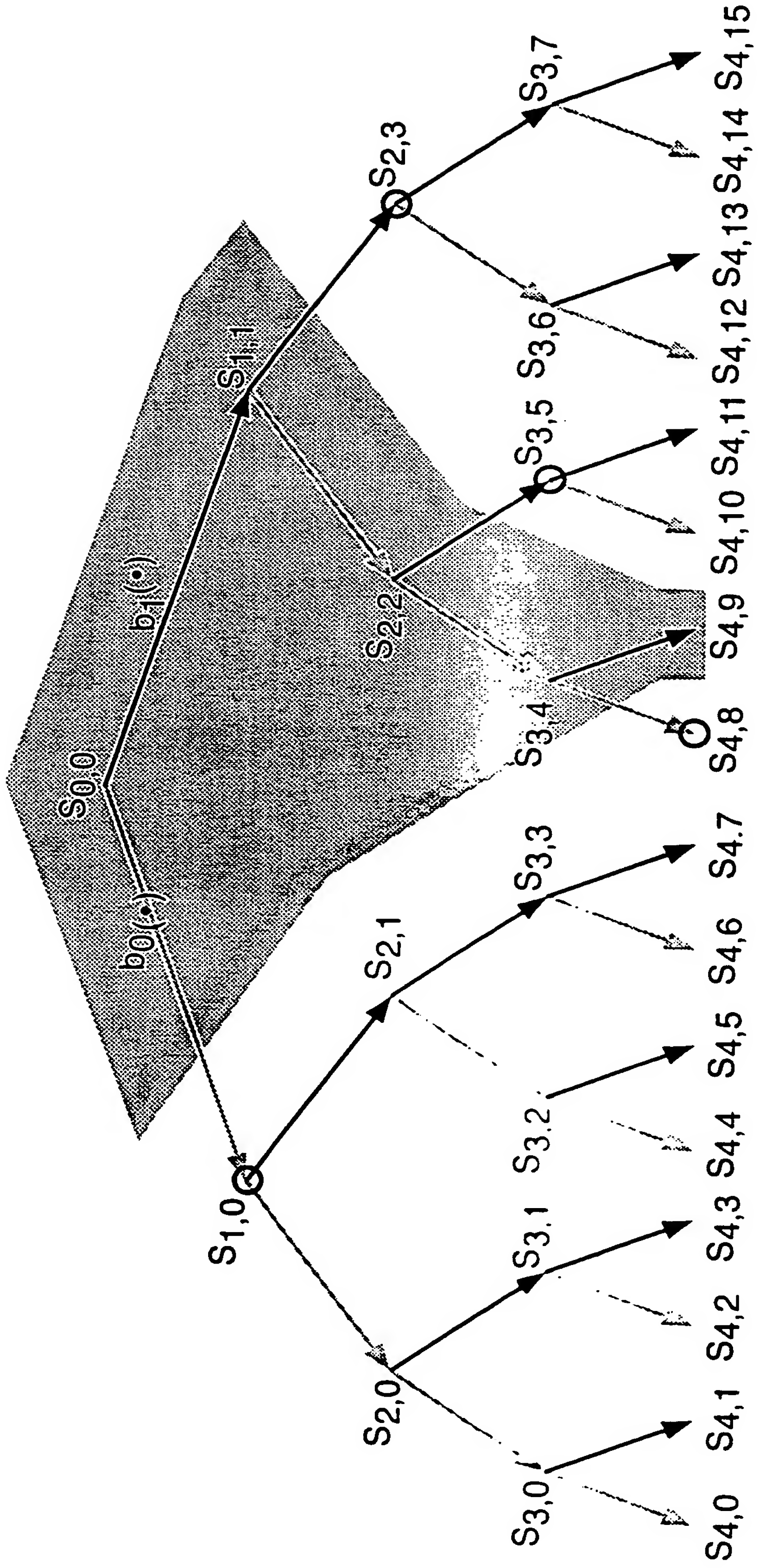
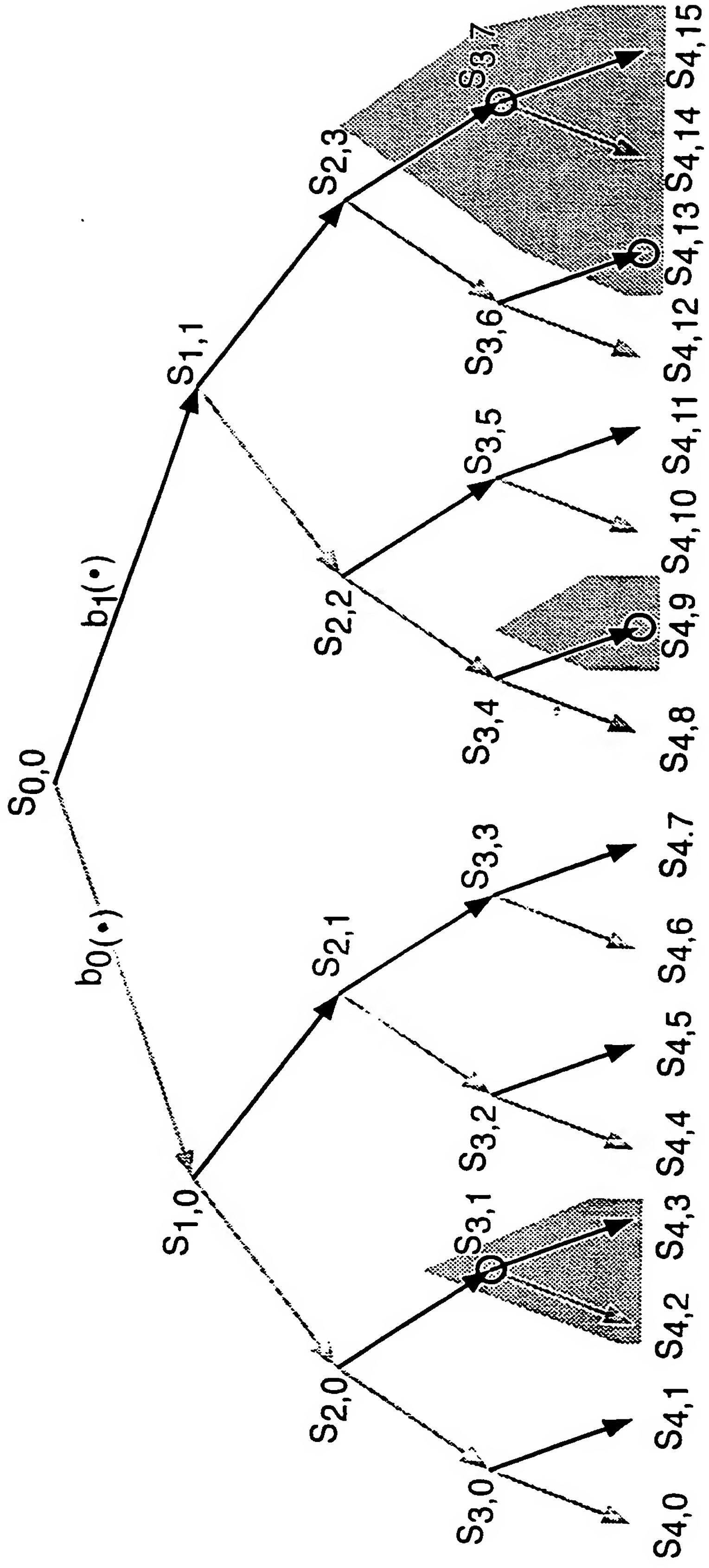

$S_{0,0}$
$b_0(\bullet)$
$b_1(\bullet)$
$S_{1,0}$
$S_{1,1}$
$S_{2,0}$
$S_{2,1}$
$S_{2,2}$
$S_{2,3}$
$S_{3,0}$
$S_{3,1}$
$S_{3,2}$
$S_{3,3}$
$S_{3,4}$
$S_{3,5}$
$S_{3,6}$
$S_{3,7}$
$S_{4,0}$
$S_{4,1}$
$S_{4,2}$
$S_{4,3}$
$S_{4,4}$
$S_{4,5}$
$S_{4,6}$
$S_{4.7}$
$S_{4,8}$
$S_{4,9}$
$S_{4,10}$
$S_{4,11}$
$S_{4,12}$
$S_{4,13}$
$S_{4,14}$
$S_{4,15}$

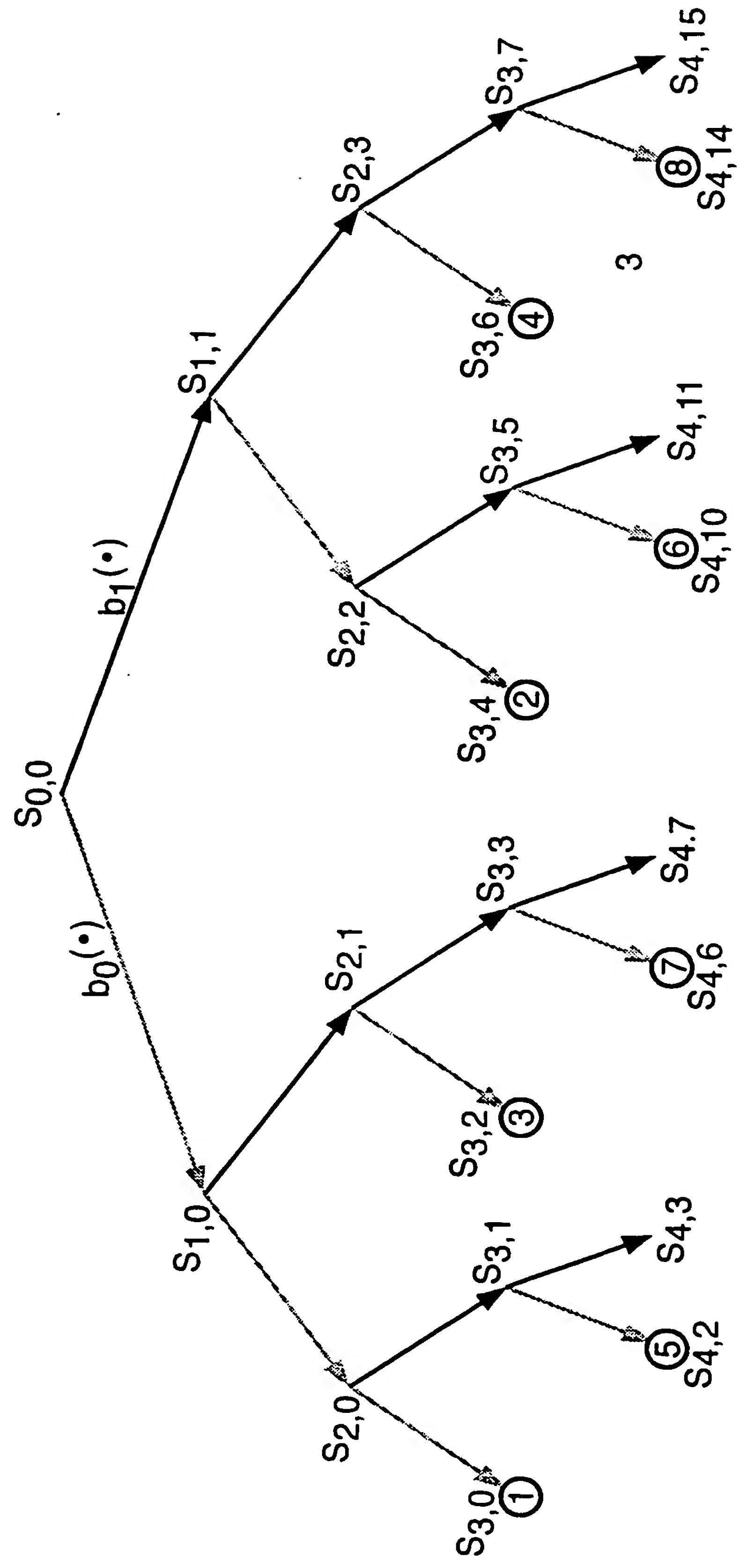
Fig.3.



$$S_e = S_{3,1} \oplus S_{4,9} \oplus S_{4,13} \oplus S_{3,7};$$

$$K_{i+1} = b(K_i, S_e)$$

Fig.4.


$S_{0,0}$
$b_1(\bullet)$
$b_0(\bullet)$
$S_{1,1}$
$S_{1,0}$
$S_{2,3}$
$S_{2,2}$
$S_{2,1}$
$S_{2,0}$
$S_{3,7}$
$S_{3,6}$ 4
$S_{3,5}$
$S_{3,4}$ 2
$S_{3,3}$
$S_{3,2}$ 3
$S_{3,1}$
$S_{3,0}$ 1
$S_{4,15}$
8 $S_{4,14}$
3
$S_{4,11}$
6 $S_{4,10}$
$S_{4.7}$
7 $S_{4,6}$
$S_{4,3}$
5 $S_{4,2}$

INTERNATIONAL SEARCH REPORT

Internat 'l Application No
PCT/GB 01/00765

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 H04L9/08

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, INSPEC, WPI Data, IBM-TDB

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	"KEY ESTABLISHMENT IN LARGE DYNAMIC GROUPS USING ONE-WAY FUNCTION TREES" AVAILABLE FROM INTERNET <URL:http://www.cs.umbc.edu/{sherman> 20.12.1999" D. MCGREW, A.SHERMAN, 20 May 1998 (1998-05-20), pages 1-13, XP002126220 cited in the application abstract page 2, last paragraph -page 3, line 20 page 10, line 23 - last line --- -/--	1

☒ Further documents are listed in the continuation of box C.

□ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search 11 June 2001	Date of mailing of the international search report 19/06/2001
Name and mailing address of the ISA European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Authorized officer Holper, G

INTERNATIONAL SEARCH REPORT

Internal al Application No
PCT/GB 01/00765

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	DINSMORE P T ET AL: "Policy-based security management for large dynamic groups: an overview of the DCCM project" PROCEEDINGS DARPA INFORMATION SURVIVABILITY CONFERENCE AND EXPOSITION. DISCEX'00, PROCEEDINGS DARPA INFORMATION SURVIVABILITY CONFERENCE AND EXPOSITION. DISCEX'00, HILTON HEAD, SC, USA, 25-27 JAN. 2000, pages 64-73 vol.1, XP000911141 1999, Las Alamitos, CA, USA, IEEE Comput. Soc, USA ISBN: 0-7695-0490-6 page 68, right-hand column, line 31 -page 70, left-hand column, line 16	1
A	CHANG I ET AL: "Key management for secure Internet multicast using Boolean function minimization techniques" IEEE INFOCOM '99. CONFERENCE ON COMPUTER COMMUNICATIONS. PROCEEDINGS. EIGHTEENTH ANNUAL JOINT CONFERENCE OF THE IEEE COMPUTER AND COMMUNICATIONS SOCIETIES. THE FUTURE IS NOW (CAT. NO.99CH36320), PROCEEDINGS OF INFOCOM'99: CONFERENCE ON COMPUTER COMMU, pages 689-698 vol.2, XP000911139 1999, Piscataway, NJ, USA, IEEE, USA ISBN: 0-7803-5417-6 cited in the application page 690, left-hand column, last paragraph -page 694, left-hand column, paragraph 1	1